

Problem: Connected Roads

ณ ประเทศอันลึกลับแห่งหนึ่งที่มีผู้นำประเทศได้คิดนโยบายเพื่อแก้ปัญหารถติดโดยการสร้างถนนเชื่อมต่อไปยังเมืองต่าง ๆ แบบทางเดียว (ถนนแบบ one-way) ทั่วทั้งประเทศ ซึ่งทำให้การเดินทางระหว่างเมืองหนึ่งสามารถใช้ถนนเส้นหนึ่งแต่ไม่สามารถใช้ถนนเส้นเดิมเดินทางกลับได้ โดยมีนโยบายให้แต่ละเมืองใช้เงินภาษีของแต่ละเมืองในการก่อสร้างถนนทั้งหมดที่สามารถออกเดินทางจากเมืองนั้นได้ เมื่อทำการก่อสร้างถนนทั้งหมดเสร็จเรียบร้อยแล้ว ผู้ปกครองประเทศอยากทราบว่าแต่ละเมืองจะใช้งบประมาณเท่าไร (สมมติว่าราคาก่อสร้างถนนต่อกิโลเมตรเท่ากันทั้งหมดทุกเมือง)

ให้นักเรียนเขียนโปรแกรมเพื่อหาคำตอบให้กับผู้ปกครองประเทศดังกล่าวโดยให้เรียงลำดับเมืองที่ต้องใช้งบประมาณจากมากไปหาน้อย ถ้าหากใช้งบประมาณเท่ากันให้เรียงลำดับหมายเลขเมืองจากน้อยไปหามาก

INPUT

บรรทัดแรก ประกอบด้วยตัวเลขจำนวนเต็ม 2 จำนวน (N แทน จำนวนเมืองทั้งหมด และ M แทนจำนวนถนนทั้งหมด) $1 \leq N \leq 20$, $1 \leq M \leq 100$

M บรรทัดถัดมา ประกอบด้วยเลขจำนวนเต็ม A B และ D แทนเมืองต้นทาง เมืองปลายทาง และระยะทางระหว่างสองเมือง (กิโลเมตร)

OUTPUT

แสดงหมายเลขเมืองทั้งหมดโดยเรียงลำดับเมืองที่ต้องใช้งบประมาณจากน้อยไปหามาก ถ้าหากใช้งบประมาณเท่ากันให้เรียงลำดับหมายเลขเมืองจากน้อยไปหามากเช่นกัน

SAMPLE TEST

Input 5 6 2 1 3 2 3 4 3 4 4 3 5 4 5 3 1	Input 5 10 1 2 3 1 4 2 1 5 3 2 1 2 2 3 3 2 4 5 3 4 2 3 5 5 5 4 4 5 3 2
Output 1 0 4 0 5 3 2 7 3 8	Output 4 0 5 6 3 7 1 8 2 10

Expression Tree

Expression Tree คือ Binary Tree ที่ใช้เก็บนิพจน์ทางคณิตศาสตร์ โหนดของ Expression Tree ประกอบด้วยข้อมูล 2 แบบ คือ

1. Operand คือ ตัวแปร หรือค่าคงที่ที่ถูกดำเนินการทางคณิตศาสตร์ เช่น a, b, x, y หรือ 1, 2, 10, 50 เป็นต้น โดย Operand จะถูกเก็บไว้ที่โหนดใบ
2. Operator คือ เครื่องหมายที่ใช้คำนวณ เช่น +, -, *, / เป็นต้น โดย Operator จะถูกเก็บไว้ที่โหนดภายใน

ข้อสังเกต Sub Tree เป็นนิพจน์ย่อยที่มี Root Node หรือ Parent Node เป็น Operator เสมอ ตัวอย่างดังรูป

จากรูป นิพจน์ทางคณิตศาสตร์ คือ $a * (b + c) - d$

นิพจน์ย่อย คือ $b + c$ และ $a * (b + c)$ (ในกรอบสามเหลี่ยม)

จงเขียนโปรแกรมเพื่อหาผลลัพธ์ของนิพจน์ทางคณิตศาสตร์จาก Expression Tree ที่กำหนดให้

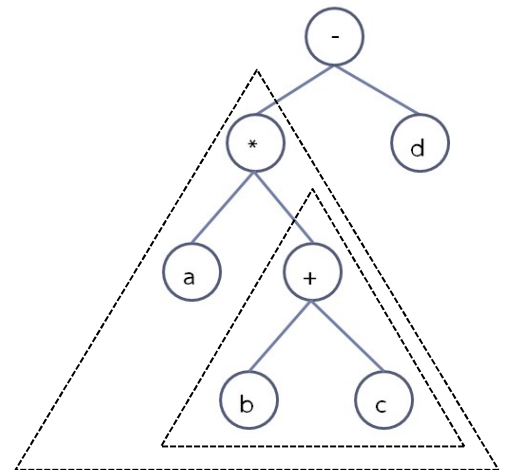
Input : บรรทัดที่ 1 คือ ขนาดของอาร์เรย์ที่ใช้เก็บ Expression Tree
บรรทัดที่สอง คือ ข้อมูลแต่ละตัวจากอาร์เรย์ (เว้นวรรคข้อมูลแต่ละตัว) โดยในที่นี้กำหนดให้

- Operator ประกอบด้วยเครื่องหมาย +, -, * และ / เท่านั้น
- Operand คือ จำนวนเต็มบวก
- หากข้อมูลมีค่าเท่ากับ -1 หมายถึงไม่มีโหนด ณ ตำแหน่งนั้น จึงไม่ต้องนำมาใช้คำนวณ

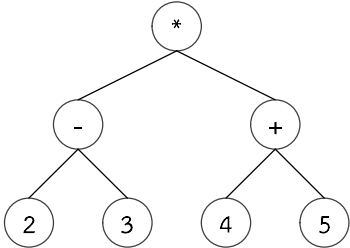
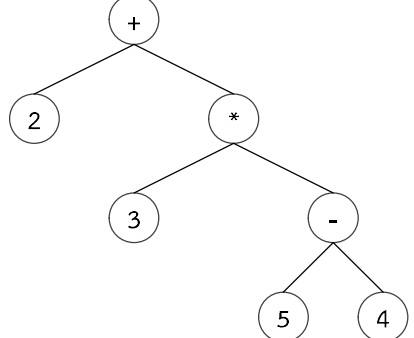
ตัวอย่างเช่น {-, *, d, a, +, -1, -1, -1, -1, b, c} คือ อาร์เรย์ที่ใช้เก็บ Expression Tree ของรูปข้างต้น

Output : ผลลัพธ์ที่ได้จากการคำนวณของ Expression Tree (ทศนิยม 2 ตำแหน่ง)

Sample :

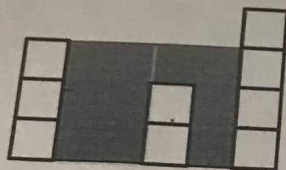


Input	Output	Note
11 + - 5 2 * -1 -1 -1 -1 3 4	-5	

7 * - + 2 3 4 5	-9	
15 + 2 * -1 -1 3 - -1 -1 -1 -1 -1 5 4	5	

Waterlogging

จงเขียนโปรแกรมเพื่อหาปริมาณน้ำที่ถูกขังไว้ระหว่างบล็อก โดยการนำบล็อกที่มีความสูงและความหนาบล็อกละ 1 หน่วยมาวางตามแนวแกน x ตัวอย่างเช่น



(a)



(b)

จากรูป (a) จำนวนบล็อกที่นำมาขังน้ำ คือ 3, 0, 0, 2, 0, 4
ปริมาณน้ำที่ขังได้คือ 10 หน่วย

จากรูป (b) จำนวนบล็อกที่นำมาขังน้ำ คือ 2, 0, 2 ปริมาณน้ำที่
ขังได้คือ 2 หน่วย

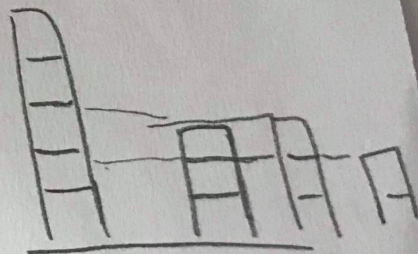
จงเขียนโปรแกรมเพื่อหาปริมาณน้ำที่ถูกขังระหว่างบล็อก

Input: บรรทัดที่ 1 คือ ขนาดของอาร์เรย์ที่ใช้เก็บจำนวนบล็อก

บรรทัดที่ 2 คือ จำนวนบล็อกที่ใช้ในแต่ละช่วง (เว้นวรรคข้อมูลแต่ละตัว)

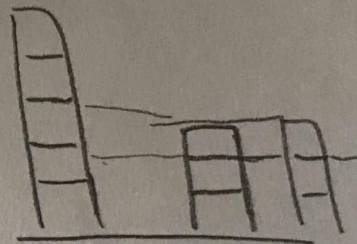
Output: ปริมาณน้ำที่ถูกขัง

Sample:



Input	Output	Note
6 3 0 0 2 0 4	10	รูป (a)

Input: บรรทัดที่ 1 คือ ขนาดของอาร์เรย์ที่ใช้เก็บจำนวนบล็อก
 บรรทัดที่ 2 คือ จำนวนบล็อกที่ใช้ในแต่ละช่วง (เว้นวรรคข้อมูลแต่ละตัว)
 Output: ปริมาณน้ำที่ถูกขัง
 Sample:



Input	Output	Note
6 3 0 0 2 0 4	10	รูป (a)
3 2 0 2	2	รูป (b)
6 1 3 2 1 2 1	1	
8 1 0 3 0 2 1 2 1	4	